Kritik Assignment #9

1a) $f(x,y) = \dfrac{x^2 - y^2}{x^2 + y^2}$

Assume $m = 1$

$$\lim_{(x,y)\to(x,mx)} f(x,y) = \frac{x^2 - (mx)^2}{x^2 + (mx)^2}, \quad = \frac{x^2 - (2x)^2}{x^2 + (2x)^2}$$

$$= \frac{x^2 - x^2}{x^2 + x^2} \qquad = \frac{x^2 - 4x^2}{x^2 + 4x^2}$$

$$= 0/2x^2 \qquad = \frac{\cancel{x^2}(1-4)}{\cancel{x^2}(1+4)}$$

$$= \emptyset \qquad = \frac{-3}{5}$$

b) $$\lim_{(x,y)\to(x,x^2)} f(x,y) = \frac{x^2 - (x^2)^2}{x^2 + (x^2)^2}$$

$$= \frac{x^2 - x^4}{x^2 + x^4}$$

sub-in $\emptyset$ for x

$$= \frac{\cancel{x^2}(1 - x^2)}{\cancel{x^2}(1 + x^2)}$$

$$= 1$$

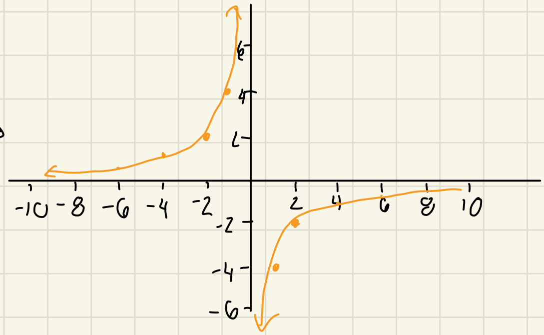
c) Based upon the results above the limit as (x,y) approaches $(0,0)$ does not exist as the limit is not equal for any parameterized curve

d) $f(x,y)$ is most likely not continuous at $(0,0)$ as we gain different values depending on how we approach $(0,0)$

2a) ① $g(x,y) = -4$

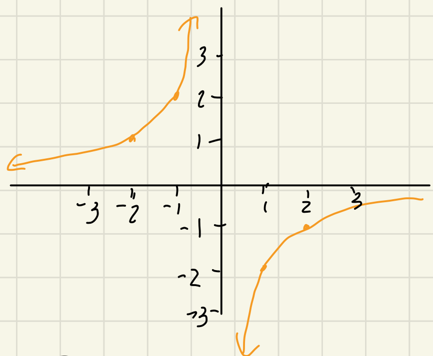
$$xy = -4$$

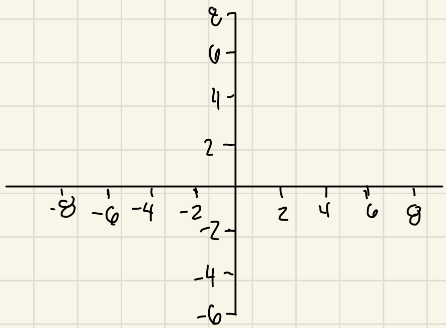
$$y = \frac{-4}{x}$$



② $xy = -2$

$$y = \frac{-2}{x}$$





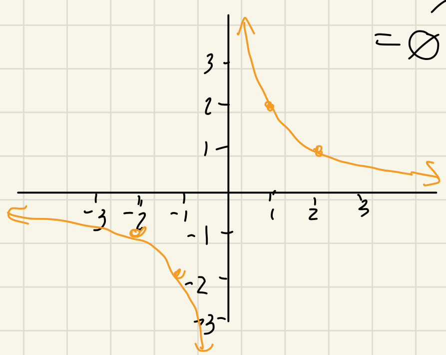
③ $xy = \emptyset$

$$y = \frac{\emptyset}{x}$$

$$= \emptyset$$

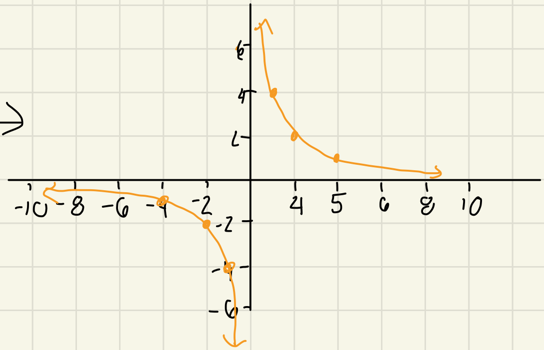
④ $xy = 2$

$$y = \frac{2}{x}$$

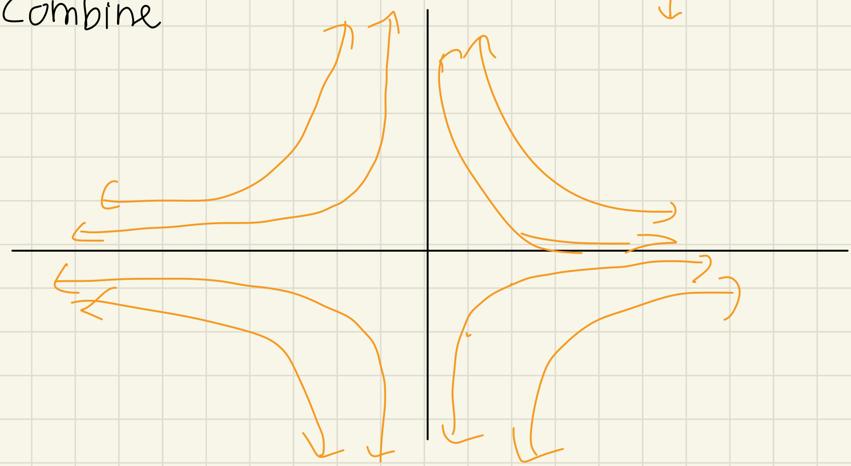


⑤ $xy = 4$

$y = \frac{4}{x}$ $\longrightarrow$



⑥ Combine



b) The spacing between level curves reflects the rate of change of g(x,y) as if increasing z by the same interval each time and the level curves become closer together it suggests that the function is becoming steeper The opposite also holds true. If level curves become more spread apart it suggest the rate of change is lower

c) Level curves can help reflect regions with maximum or minimum values in a function as when a 3-D plane nears a max value the level curves will often come closer together or form a circle

3a) $\frac{\partial h}{\partial x} = e^{x^2+y^2} \cdot 2x$

$\frac{\partial h}{\partial y} = e^{x^2+y^2} \cdot 2y$

b) $\frac{\partial h}{\partial x} = e^{(1)^2 + (-1)^2} \cdot 2(1)$
$= 2e^2$

$\frac{\partial h}{\partial y} = e^{(1)^2 + (-1)^2} \cdot 2(-1)$
$= -2e^2$

c) $\frac{\partial^2 h}{\partial x^2} = \frac{\partial h}{\partial x} e^{x^2+y^2} \cdot 2x$

$= e^{x^2+y^2} \cdot 2x \cdot 2x + 2e^{x^2+y^2}$

$= 4x^2 e^{x^2+y^2} + 2e^{x^2+y^2}$

$\frac{\partial^2 h}{\partial y^2} = \frac{\partial h}{\partial y} e^{x^2+y^2} \cdot 2y$

$= e^{x^2+y^2} \cdot 2y \cdot 2y + 2e^{x^2+y^2}$

$= 4y^2 e^{x^2+y^2} + 2e^{x^2+y^2}$

$\frac{\partial^2 h}{\partial x \partial y} = \frac{\partial h}{\partial y} e^{x^2+y^2} \cdot 2x$

$= e^{x^2+y^2} \cdot 2y \cdot 2x = 2x \cdot 2y \cdot e^{x^2+y^2}$

$$\frac{\partial^2 h}{\partial y \partial x} = \frac{\partial h}{\partial x} \, 2y e^{x^2+y^2}$$

$$= 2y \cdot 2x \cdot e^{x^2+y^2}$$

$$\frac{\partial^2 h}{\partial y \partial x} = \frac{\partial^2 h}{\partial x \partial y}$$ } can differentiate in either order and will gain the same value!

d) Both x & y have a derivative of either $2x e^{x^2+y^2}$ or $2y e^{x^2+y^2}$. This means they both change by the same magnitude along their respected axis. However at point $(1,-1)$ this means that the x-axis is traveling to positive direction while the y-axis travels to negative direction.

4. Python can be seen below or on file attached

```python
def gradient_descent(x0=0.1, y0=0.1, alpha=0.1, num_iterations=10):
    """
    Parameters:
    x0, y0: Initial point for the descent.
    f: a function of two variables
    grad_f: the gradient of f
    alpha: Learning rate.
    num_iterations: Number of iterations to perform.
    Returns:
    (x, y): The coordinates of the final point after gradient descent.
    """
    x, y = x0, y0 # Initialize x and y with the initial point



    for i in range(num_iterations):
        # obtain the gradient of f at (x, y)
        grad_x, grad_y = 2*x, 2*y
        # Update x and y by taking a step in the

        # opposite direction of the gradient
        x = x-alpha*grad_x
        y = y-alpha*grad_y

    return x, y
```

```python
x_y = gradient_descent(x0=0.1, y0=0.1, alpha=0.1, num_iterations=10)
print(x_y)
```

```
(0.010737418240000003, 0.010737418240000003)
```

```python
x_y = gradient_descent(x0=-1, y0=1, alpha=0.01, num_iterations=100)
print(x_y)
```

```
(-0.13261955589475316, 0.13261955589475316)
```

```python
import numpy as np
from mpl_toolkits import mplot3d #for 3D plots
import matplotlib.pyplot as plt #usual matplotlib


def gradient_descent_2(x0=0, y0=1, alpha=0.01, num_iterations=1000):
    """
    Parameters:
    x0, y0: Initial point for the descent.
    f: a function of two variables
    grad_f: the gradient of f
    alpha: Learning rate.
    num_iterations: Number of iterations to perform.
    Returns:
    (x, y): The coordinates of the final point after gradient descent.
    """
    x, y = x0, y0 # Initialize x and y with the initial point



    for i in range(num_iterations):
        # obtain the gradient of f at (x, y)
        grad_x = -(np.exp(-x**2-(y-2)**2)*(-2*x))-2*(np.exp(-x**2-(y+2)**2)*(-2*x)) #Derivatice with respect to x
        grad_y = -(np.exp(-x**2-(y-2)**2)*(-2*(y-2)))-2*(np.exp(-x**2-(y+2)**2)*(-2*(y-2))) #Derivative with respect to y
        # Update x and y by taking a step in the

        # opposite direction of the gradient
        x = x-alpha*grad_x
        y = y-alpha*grad_y

    return x, y
```

```python
x_y = gradient_descent_2(x0=0, y0=1, alpha=0.01, num_iterations=1000)
print(x_y)
```

```
(0.0, 1.9999999966926432)
```

```python
x_y = gradient_descent_2(x0=0, y0=-1, alpha=0.01, num_iterations=1000)
print(x_y)
```

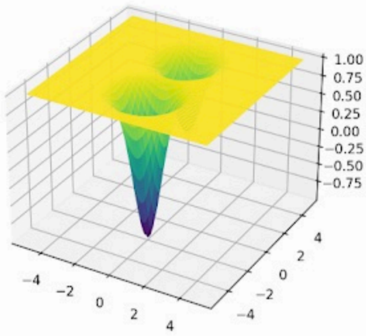
```
(0.0, 1.9999581429124287)
```

```
import numpy as np
from mpl_toolkits import mplot3d #for 3D plots
import matplotlib.pyplot as plt #usual matplotlib
%matplotlib widget

X=np.linspace(-5,5,100)
Y=np.linspace(-5,5,100)
x,y=np.meshgrid(X,Y)
z= 1-np.exp(-x**2-(y-2)**2)-2*np.exp(-x**2-(y+2)**2) #One of the functions given by Greg

fig = plt.figure()
ax = plt.axes(projection='3d')
ax.plot_surface(x, y, z,cmap='viridis', edgecolor='none')
#x,y z are variable names.
```

Figure 5




```
import numpy as np
from mpl_toolkits import mplot3d #for 3D plots
import matplotlib.pyplot as plt #usual matplotlib
%matplotlib widget

X=np.linspace(-5,5,100)
Y=np.linspace(-5,5,100)
x,y=np.meshgrid(X,Y)
z= x**2+y**2 #The other function given by Greg

fig = plt.figure()
ax = plt.axes(projection='3d')
ax.plot_surface(x, y, z,cmap='viridis', edgecolor='none')
#x,y z are variable names.
```

```
<mpl_toolkits.mplot3d.art3d.Poly3DCollection at 0x7f17600cb6d0>
```

Figure 2